"Express Mail" mailing label number EV 327 130 016 US

Date of Deposit: July 21, 2003

Patent 2003P50286US
Our Case No. 10808/101

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
APPLICATION FOR UNITED STATES LETTERS PATENT

INVENTORS:	Gerd Frankowsky
TITLE:	MEMORY DEVICE AND METHOD OF STORING FAIL ADDRESSES OF A MEMORY CELL
ATTORNEY:	John J. King (Reg. No. 35,918) BRINKS HOFER GILSON & LIONE POST OFFICE BOX 10395 CHICAGO, ILLINOIS 60610 (312) 321-4200

MEMORY DEVICE AND METHOD OF

STORING FAIL ADDRESSES OF A MEMORY CELL

FIELD OF THE INVENTION

[0001] The present invention relates generally to memory devices, and in particular, to a memory device and method of storing fail addresses of a memory cell.

BACKGROUND OF THE INVENTION

[0002] Memory tests on semiconductor devices, such as random access memory devices, are typically performed by the manufacturer during fabrication to locate defects and failures in the devices that can occur during the fabrication of the device. This testing is typically performed by a processor which runs a testing program before the die containing the semiconductor device is packaged into a chip. Many semiconductor devices including memory devices typically have redundant circuitry to replace malfunctioning circuitry found during testing. By enabling the redundant circuitry, the device need not be discarded even if some of the circuits are defective.

[0003] As memory size increases, the time required to test a DRAM being manufactured must increase as well, resulting in additional cost. Also, as DRAM applications grow and processors begin to incorporate large quantities of memory on chip, it is necessary to create new testing schemes for these embedded memories. When there is no direct connection between pins and memory, external

testing can be difficult. That is, ensuring that every bit is tested thoroughly requires complex test vectors.

[0004] Built in self testing (BIST) Built in self testing enables cheaper testing equipment to be used to test the memory device, as well as parallel testing which can increase chip production rates. Finally, built in self tests can, in some cases, be performed throughout the operational life of the chip. However, conventional methods for determining defects in a semiconductor device are complex and can be time consuming.

[0005] Accordingly, there is a need for a memory device and method of determining defects in a semiconductor device by scanning the memory array to locate the word lines and column select lines with the highest number of defects and by storing fail addresses of a memory cell.

BRIEF SUMMARY OF THE INVENTION

[0006] The embodiments of the present invention are directed to a self-repair schema for memory chips using a sortable fail-count/fail-address register. The embodiments of the present invention utilize the available redundancy efficiently by scanning the memory array to locate the n elements (WLs or CSLs) with the highest number of defects. A circuit preferably comprises one or more comparators to compare a fail count of an address in an input register with at least one fail count stored in the sortable fail-count / fail-address register. The embodiments of the present invention can be used for an on-chip redundancy calculation and can handle a two dimensional (i.e. row and column) redundancy.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] Fig. 1 is a block diagram of memory employing the embodiments of the present invention;

[0008] Fig. 2 is a block diagram of a circuit for storing failure addresses of a memory cell according to the present invention;

[0009] Fig. 3 is a block diagram of a circuit for storing failure addresses of a memory cell according to an alternate embodiment of the present invention;

[0010] Fig. 4 is a diagram showing the operation of the circuit of Fig. 3;

[0011] Fig. 5 is a flow chart showing a method of scanning word lines and column select lines according to an embodiment of the present invention;

[0012] Fig. 6 is a flow chart showing a method of scanning cells of a memory device according to an embodiment of the present invention;

[0013] Fig. 7 is a flow chart showing a method of storing fail addresses in a register according to an embodiment of the present invention; and

[0014] Fig. 8 is a flow chart showing a method of storing fail addresses in a register according to an alternate embodiment of the present invention.

DETAILED DESCRIPTION OF THE DRAWINGS

[0015] Turning first to Fig. 1, a block diagram of memory device such as a dynamic random access memory (DRAM) employing the embodiments of the present invention is shown. In particular, a memory device 100 comprises a plurality of arrays 102, each of which comprises a memory array bank 104, a row decoder 106, a sense amp and input/output bus 108, and a column decoder 110. The memory device

100 further comprises a row address register 120 coupled to a row address buffer 122, which receives a signal from a refresh counter 124. Similarly, a column address register 130 is coupled to a column address buffer 132 which provides an output to a column address counter 134. A data input/output block 140 is coupled to an input buffer 142 and an output buffer 144. The input buffer 142 and the output buffer 144 enable the transfer of data to and from the various arrays. Finally, a control logic and timing generator block 150 receives control signals, which are well known in the art, for reading or writing data to the memory device.

[0016] Turning now to Fig. 2, a block diagram of a circuit for storing failure addresses of a memory cell according to the present invention is shown. The stack of Fig. 2 contains n registers, and could be implemented, for example, in the control logic and timing generator block 150. Each register is divided into two parts, one to store the fail bit address and the other to store the fail-count. The bits containing fail count information of each register are further connected to a comparator. A stack with a depth of n registers therefore requires n comparators.

[0017] In particular, circuit 200 comprises an input register 202 having a fail address input portion 204 and a fail count portion 206. The circuit 200 also comprises a plurality of registers 210. Each register 210 comprises a fail address portion 212 and a fail count portion 214. A comparator 216 receives the fail count portion 214 and fail count portion 206 of the input register 202. A switch 218, which is controlled by the output of the comparator 216, enables the transfer of a fail address and a failed count to the input register 202. Finally, a clock signal 220 is coupled to a switch 222, enabling the transfer of data on the falling edge of the clock signal. The clock signal is also coupled to the input of the comparator 216 to enable the comparison on the

rising edge of the clock signal. As can be seen in the circuit, the comparator is enabled on the rising of the clock signal, generating an output which is coupled to the switch 218. The state of the switch 18 determines whether the fail address and the fail count are transferred from the input register to one of the registers 210. The operation of the circuit will be described in more detail in reference to the flowchart of Fig. 7.

[0018] One advantage of the implementation of Fig. 2 is the speed in determining defects in the semiconductor device. The new fail-count can be simultaneously compared to all fail-counts stored on the register at the rising clock edge. Depending on the compare results, the new fail-address and fail-count can be transferred to the stack at the falling clock edge. Therefore adding a new element to the stack can be done within one clock cycle. The drawback of the solution of Fig. 2 is the required chip area, because each stack element requires a comparator.

[0019] Turning now to Fig. 3, a block diagram of a circuit for storing failure addresses of a memory cell according to an alternate embodiment of the present invention is shown. Because the embodiment of Fig. 3 comprises only one comparator, the fail count in the input register is only compared against the fail count stored in register R[0]. The register stack can be rotated so that the input fail count can be compared to the fail counts stored in all registers. Two rotate operations are possible. The first rotate would be based upon fail address stored in the register stack only, while a second rotate would be based upon the register stack and the input register.

[0020] In particular, a circuit 300 comprises an input register 302 having an input fail address 304 and an input fail count 306. The circuit 300 comprises a register stack 310 having a plurality of fail address registers 312 and fail count registers 314.

A comparator 320 compares a fail count of register R[0] with the input fail count 306. Finally, a switch 322 enables the transfer of the input fail address 304 and the corresponding input fail count 306 into the register R[n-1]. One advantage of the circuit of Fig. 3 is that the required area for this circuit is smaller compared to Fig. 4 because only one comparator is required. However, adding a new element to the stack takes longer. As be described in more detail in reference to Fig. 4, n+l clock cycles are required to add a new element for a stack with a depth of n elements.

[0021] Turning now to Fig. 4, a diagram shows the operation of the circuit of Fig. 3 for a register stack having a depth of 4 registers. At the beginning of the operation, the new fail count is stored in the input register (I). As long as the new fail count is lower than the fail count stored in the Register R[0], a 'stack-only' rotation is performed, as shown for example in clock cycles 0 and 1. That is, the fail address and fail count stored in register R[0] are shifted to the register R[n-1], while the fail addresses and the fail counts stored in the other registers are shifted down as shown. However, once the comparator detects a lower fail count on the stack, the rotation-mode is changed. In particular, a rotation of both the input register and the registers in the stack is performed. The operation ends after n+l clock-cycles. At that point, the register stack having the lowest fail count is in the input register, where it will be overwritten at the beginning of the next stack-operation.

[0022] Turning now to Fig. 5, a flow chart shows a method of scanning word lines and column select lines according to an embodiment of the present invention. In particular, word lines are scanned several times in opposite directions at a step 502. An example of a method of scanning the word lines is described in more detail in reference to Fig. 6. The column select lines are also scanned several times in opposite

directions at a step 504. It will be understood that the column select lines could be scanned according to the method of Fig. 6 as applied to columns rather than rows. Word lines or column lines are then replaces with redundant elements at a step 506. It is then determined whether all redundant elements have been used at a step 508. If so, the process is ended. If not, it is then determined whether any more defects are found at a step 510.

[0023] Turning now to Fig. 6, a flow chart shows a method of scanning cells of a memory device according to an embodiment of the present invention. For the following discussion we will assume that X represents the row address and Y the column address. Accordingly, the repair strategy is to find the n wordlines with the highest fail counts. In particular, the register stack is cleared at a step 602. The row- and column-address counters are set to their start values (e.g. X=0, Y=0) at steps 604 and 606, respectively.

[0024] According to one aspect of the present invention, all cells along the wordline X are scanned by increment the column address Y. The fail count is reset to zero at the step 608. The cell (X,Y) is tested at a step 610. In case the cell shows a defect at a step 612, the fail-bit counter is incremented at a step 614. This is done for all memory cells on the wordline. After all cells on the first wordline have been tested at a step 618, the fail-count F and the address X are transferred to the fail-address stack at a step 620. The fail address stack accepts a new address and fail-count only if at least one element on the stack has a lower fail count, as will be described in more detail in reference to Figs. 7 and 8. The row is then incremented at a step 622. This operation is repeated until all wordlines have been tested, as determined at a step 624. Now, the fail address stack contains the addresses of the n

wordlines with the highest fail counts. These wordlines are replaced now by activating redundant elements at a step 626.

[0025] In order to repair the chip, it is preferable to run the algorithm several times in opposite directions. This means that after scanning the wordlines, one would use the same algorithm to scan the CLSs. This procedure is repeated until no more defect cells are found or all redundancy elements have been used. One advantage of this method is that the redundancy is used very efficiently.

[0026] Turning now to Fig. 7, a flow chart shows a method of storing fail addresses in a register according to an embodiment of the present invention. In particular, a new fail address and fail count are loaded into an input register at a step 702. The new fail count is simultaneously compared to all fail counts stored in a stack register at a step 704. It is then determined whether the new fail count is greater than a stored fail count at a step 706. If so, the stored address and corresponding fail count in the register stack is then replaced with the new address and register count from the input register at a step 708. It is then determined if all fail addresses have been loaded in the register stack at a step 710. If not, a new fail address and fail count are loaded into an input register at the step 702. However, determined if all fail addresses have been loaded in the register stack, the rows (or columns) associated with the stored addresses are replaced with redundant elements at a step 712.

[0027] Finally, turning to Fig. 8, a flow chart shows a method of storing fail addresses in a register according to an alternate embodiment of the present invention. In particular, a new fail address and fail count is loaded into an input register at a step 802. The fail count in the input register is compared to the fail count in the top slot of the register stack at a step 804. It is then determined whether the new fail count is

greater than the fail count in the top slot at a step 806. If not, the address and the fail count in the top slot is shifted to the bottom slot at a step 808. If so, the new fail count is shifted to the bottom of the stack at a step 810. The fail address and corresponding fail count is then shifted to the input register at a step 812. It is then determined if the input register has been compared to the top slot n+1 times at a step 814. If not, the fail count in the input register is compared to the fail count in top slot of the register stack at the step 804. If so, the rows (or columns) associated with the stored addresses are replaced with redundant elements at a step 816. It should be understood that the reference to top slot and bottom slot are merely given by way of example (as shown for example in Fig. 4).

[0028] It can therefore be appreciated that the new and novel memory device and method of storing failure addresses of a memory cell has been described. It will be appreciated by those skilled in the art that, particular the teaching herein, numerous alternatives and equivalents will be seen to exist which incorporate the disclosed invention. As a result, the invention is not to be limited by the foregoing embodiments, but only by the following claims.